EXHIBIT 7

(PART 1 OF 4)

RETURN TO: M.I. GASPER

5216 STANDARD ~~FIRMWARE~~ LIST

USER'S MANUAL

Document No. 150-6045-004

Revision June 1981

This manual supercedes original
document printed April 1980.

LIST OF CHANGE PAGES

Page Number	Date of Latest Change

PREFACE

The 5216 Standard Firmware has been designed to emulate the AYDIN 5214A Display Generator. Instructions which are not part of the 5214A instruction set are indicated by *EXT.

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

Section Page

APPENDICES

INDICES

LIST OF FIGURES

LIST OF TABLES

SECTION 1

GENERAL INFORMATION

The AYDIN 5216 Display Computer is a sophisticated computer whose processing ability is built around the powerful Intel 8086 16-bit microprocessor. The Standard Firmware provided with the 5216 Display Computer has an instruction set of largely alphanumeric and graphic instructions which accepts and processes instructions for display.

The Standard Firmware is a very powerful display instruction set, but it does not provide its own operating system. The 5216 Display Computer can be provided with a second processor card, with an AYDIN supplied or user supplied operating system. One processor card contains the operating system, and the other has the Standard Firmware alphanumeric and graphic display instruction set. In this configuration, the 5216 is a standalone computer.

Alternatively, the operating system may be supplied by a host computer which communicates with the 5216 as if it were a peripheral device of the host. AYDIN provides the interface hardware to communicate with the DEC PDP-11, Data General Nova or Eclipse, and other host computers.

The Standard Firmware instruction accepts codes from either a second processor or a host computer and then executes all the necessary code to generate alphanumeric or graphic data on the display monitor. This monitor may have picture resolutions of 1024 x 1024, 1024 x 768, 512 x 512, or 512 x 768. Depending on the display hardware configuration, up to 256 or 4096 colors may be displayed on the color monitor simultaneously.

In addition to computer control through coded instructions, another way the 5216 interfaces with the "real world" is through its keyboard inputs. Keyboard support includes a set or preprogrammed function keys which can execute most of the functions of the 5216 Standard Firmware instruction set. Similarly, the keyboard also has functions to display three sizes of alphanumeric character display, lines and circles, complete color control, fill, cursor movement, roll, scroll, zoom, and copy. It also has a program buffer in which keyboard function commands can be stored.

The Standard Firmware instruction set contains alphanumeric instructions which generate three sizes of characters: 5 x 7, 7 x 9, and 10 x 14. These dimensions are given by the number of picture elements wide by the number high, so that the size observed on the display screen will vary, depending on physical hardware configuration. These alphanumeric displays contain the numbers and upper and lower case letters of the English alphabet (including punctuation marks), the full Greek alphabet, and some other special symbols. To add extra power and versatility to the alphanumeric display capability, a programmable font can store up to 256 user-programmable characters in a 16 x 15 picture element matrix.

Cursor advance between characters is handled automatically by the firmware, and can be adjusted by programmable specification.

Alphanumeric display can be done in any color available to the 5216 hardware configuration. Each character is displayed in a rectangular matrix of picture elements whose color can also be controlled. Graphic instructions perform such high level display functions as drawing lines, circles, and arcs. Every picture element is addressable so that the programmer can perform the calculations to draw any shape. An image is created by down-line loading pixel data to any selected rectangular area of the display.

Besides alphanumeric and graphic display instructions, the 5216 Standard Firmware contains instructions to perform the following:

- Change the data already on the screen (Display Edit functions)
- Control the lookup table in the color display hardware (Lookup Table instructions)
- Store and execute lists of display instructions in 5216 RAM (Cache and Keyboard Buffer instructions, and Standard List instructions)

Display Edit functions can clear the screen, roll or scroll data on the screen, copy displayed data from one portion of the screen to another, or zoom displayed data to double its original size. These instructions can be performed on the full screen, or limited to any rectangular subsection of the screen.

Color displayed by the 5216 is determined in two ways: first, by the data the programmer has stored in refresh memory, and second, by the color value which the display hardware assigns to each stored data value via a programmable lookup table (LUT) located on the video display card. Several different video display cards are available for the 5216 Display Computer; the two most popular are the VID-101 and the VID-4. The LUT on the VID-101 card can contain 4096 colors simultaneously; the maximum number of colors in the VID-4 card LUT is 256.

Configurations in which the 5216 Display Computer is linked to a host computer may require data transmission from the 5216 to the host. Information such as current cursor location, current screen data, lookup table contents, or contents of some other area of 5216 memory may be requested by the host, using data transmission instructions.

Lists of display instructions can be stored and recalled using the

Load and Execute Cache Buffer instructions. Many instructions may be stored in Expansion RAM modules in which up to 768K bytes of RAM may be added to 5216 address space.

To increase the power and flexibility of this Cache Buffer capability, the Standard List instructions are optionally available as part of the 5216 Standard Firmware package. These list instructions contain counters and pointers to perform jumps and loops within the stored list of display instructions.

SECTION 2

SELECT PARAMETER SET

In the 5216 Standard Firmware instruction set, a number of commands are required to control the programming environment. Some instructions select the refresh memory channels on which data is to be written; others confine the screen display to some rectangular area. The Mode Control Word (MCW) selects a font size and determines other important factors about data display. These parameters are retained as a list in 5216 RAM.

To increase the flexibility and power of the 5216 Standard Firmware, 15 such lists can be stored. Any one of these lists can be "activated" at any time by issuing a Start of Message (SOM) or select parameter set instruction with the appropriate operand, from 0 to 14.

For instance, the user might want to set up the top part of the display in graphic Pixel mode with major channels one through six enabled to display graphics and image data, and set the bottom portion to alphanumeric Word mode, with one of the four available character fonts selected to display text. The user might want to set up a third parameter set to include the whole display with only major channels seven and eight selected to display overlay information. The programmer selects which of 15 parameter sets is active by issuing the SOM command.

Start of Message - SOM

													Parameter Set			
SOM	1	1	1	0	1	1	0	0	–	–	–	–	PS_3	PS_2	PS_1	PS_0
	15							8	7			4	3			0
			E		/		C				+			0 → F		

Hyphens indicate "don't care" bits

Each parameter set has its own set of the following:

<u>Parameter</u>	<u>Default Values</u>
- MAJOR Channel Select	All
- MINOR Channel Select	All
- MAJOR Channel Mask	0
- MINOR Channel Mask	0
- Mode Control Word	5 x 7, Replace, Alpha

Parameter	Default Values
- X - cursor position	0
- Y - cursor position	0
- X - index	0
- Y - index	0
- Foreground pixel value	FFFF
- Background pixel value	0000
- Rectangular limits	Full Screen
- Conic limits	Full Screen
- ACA values	1 right, 1 down
- KTS word	Local mode (interrupt to host not set)

Whenever a graphics command is sent from the host, the parameter set most recently selected by the SOM command is retrieved by the 5216 and the graphics command is executed.

The host is associated with parameter set 0 (identical to parameter set 15 decimal) at Power Up. The parameter set which the host addresses can be changed at any time by issuing an SOM instruction.

Each keyboard is assigned to a particular parameter set at Power Up; these assignments may be altered by the programmer. See Section 17 for more detail.

SECTION 3

RECTANGULAR LIMITS VIEWPORT

New display data can be limited to a rectangular section of the screen by specifying a "viewport." This rectangular area, or "viewport," is defined by issuing four limit commands: right, left, top, and bottom.

Load Rectangular Limits - LRL, LRR, LRT, LRB

$2^{10} = 1024$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEFT - LRL	0	1	0	1	0	0	L	L	L	L	L	L	L	L	L	L
			5													
RIGHT - LRR	0	1	0	1	1	0	R	R	R	R	R	R	R	R	R	R
			5													
TOP - LRT	0	1	1	0	0	0	T	T	T	T	T	T	T	T	T	T
			6													
BOTTOM - LRB	0	1	1	0	1	0	B	B	B	B	B	B	B	B	B	B
			6													

BINARY VALUE OF LIMIT

The ten least significant bits of the limit instructions are the binary values of the rectangular limits.

Specifying a viewport rectangle has no effect on data already displayed on the screen. Only new display entries are restricted by the viewport area.

The cursor may not be positioned outside the viewport rectangle. If a command is issued to position the cursor outside the viewport limits, an error condition will result (see LCX, LCY instructions in Section 5.) An error condition will not result immediately if newly specified rectangular limits do not encompass the current cursor location. However, as soon as a command which depends on the cursor position is issued, such as Load Alphanumeric Character (LAC), an error condition will result. An Edit Clear instruction (see Section 9) will automatically position the cursor at the upper left corner of rectangular viewport even if the cursor had been located outside the limits before the Clear was issued. Once the cursor is inside the viewport, most cursor movements will cause the cursor to wrap around within the viewport when the viewport edge is encountered.

Alphanumeric data entry using either the LAC or Block Transfer Mode (BXM) instruction will be advanced to the next line and returned to the beginning edge of the rectangular viewport when a limit is encountered. BXM graphic data will also automatically advance and wrap around at viewport limits. Since BXM in graphic Word mode writes 16 pixels at a time beginning on word boundaries, special rules applying to viewport boundaries in this case. The special boundaries are defaulted to automatically, but the programmer should be aware of these rules to ensure predictable results. For BXM Graphic Word data, the left limit is forced to the word boundary by setting the four lowest bits to 0 and the right limit is forced to the word boundary by setting the low four bits to 1. After the execution of the BXM, the original limits are automatically restored.

There are three exceptions to the rule that new display data may not be entered outside the viewport limits.

First and most important, conics are not limited by the rectangular viewport. Second, viewport limit checking is done before indexing takes place in the LAC and Load Graphic Elements (LGE) commands. Therefore, if the index bit is set in either of these instructions, the data may be indexed out of the viewport limits without causing an error condition. However, the cursor will still not be moved outside the limits. Third, because of its special characteristics, the Copy instruction is not affected by the viewport limits.

The origin of the coordinate system (X=0, Y=0) is at the upper left of the screen. Positive X is to the right and positive Y is down. Notice that the bottom limit must be a larger number than the top limit, and that the right limit must be a larger number than the left limit.

The Power Up value of the rectangular limits is the full screen.

SECTION 4

MODE CONTROL WORD

Mode Control Word - MCW

only for BXM

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	A/N	Z_1	Z_0	V	F_1	F_0	H	P	M

Alphanumeric Mode ACA Flag

Memory Input Mode Select

Font Select (A/N Mode Only) M=0

Packing Order Selection Bit for Block Transfer (A/N BXM only)

Pixel Mode 0 = word
1 = pixel

Display Mode Select - [Alphanumeric = 0
Graphic = 1]

Packing Order 0 = low then high
1 = high then low

ACA Flag 0 = Font default
1 = use ACA values

LAC
BXM } M=φ
XBXM

CUR
WPX
WPXF
WPXB } M=1
BXM
XBXM
EXC

Display Mode Selection

Instructions can be entered to record either alphanumeric or graphic data.

Alphanumeric data is entered using the Load Alphanumeric Character (LAC), Block Transfer Mode (BXM), and Extended Block Transfer Mode (XBXM) instructions. To use these instructions, the zero bit of the MCW must be set to 0.

Note

It is also possible to generate alphanumeric display by directly programming the Alphanumeric Channel Set (ANCS). The instructions for this programming are listed in user documentation for the ANCS and are not included in this document. The Select Device (SDEV) instruction is used to interface the 5216 Standard Firmware with the ANCS.

Graphic data may be entered for display when the zero bit of the MCW is set to 1. Instructions which are used to add graphic data for display are Cursor Movement (CUR), Write Pixel Values (WPX, WPXF, WPXB), Block Transfer Mode (BXM), and Extended Block Transfer Mode (XBXM). Also, vectors and circles are generated by the Execute Conic (EXC) instruction.

For Alphanumeric display, all the input bits of the MCW are relevant. In alphanumeric mode, bit 1 may be set to 0 for Word mode or 1 for Pixel mode, and bit 2 is only used for Block Transfer Mode (BXM). Bits 3 and 4 select the font; bits 5, 6, and 7 select the Memory Input Mode. Bit 8 chooses between default font spacing and user programmed spacing (see ACA instruction) for adjustable cursor advance.

For Graphic display, bits 2, 3, 4 and 8 are ignored.

Pixel or Word Mode

Bit 1 selects Word or Pixel mode. (This is particular to the 5216 Standard Firmware and is not implemented in the 5214A instruction set.)

In Word mode (bit 1 = 0), the 5216 hardware accesses sixteen adjacent pixel locations at a time. It must be used for Load Graphic Elements (LGE) and may be used for BXM graphic transfer. Word mode may be selected for LAC and BXM alphanumeric. Any instruction except Write Pixel (WPX) may be issued in Word mode. The firmware will address the appropriate pixel locations using masks.

When Word mode is selected, the input data from the Foreground Pixel Register (FPR) is automatically set to all 1's and the input data from the background register (BPR) is automatically set to all 0's. The contents of the pixel registers is not lost and will be restored as soon as Pixel mode is selected.

For alphanumeric data, Word mode is time efficient, but input color is limited to black and white (or some other two-color combination, depending on channel selection and the lookup table). Also, characters which overlap the window boundary will be aborted if generated in Word Mode. (In BXM Word mode, window boundaries are forced to word boundaries.)

When Pixel mode is selected (bit 1 = 1), data is entered one pixel at a time. Data entry is done in the Z direction using pixel values stored in the FPR and BPR. Alphanumeric data in both LAC and BXM may be entered in Pixel mode. When alphanumeric data is generated in the Pixel mode, characters which overlap window boundaries will be clipped. LGE may not be called while in Pixel mode.

Conics (vectors and circles) may be generated in either Pixel or Word mode. However, in word mode, the FPR and BPR will effectively be all 1's and all 0's, respectively.

Alphanumeric BXM Unpacking Order

This bit (bit 2) is only relevant for the alphanumeric BXM and Extended Block Transfer Mode (XBXM) instructions.

When bit 2 equals one (1), the data words following the BXM and XBXM instructions are unpacked with the eight most significant bits first and the eight least significant bits last.

When bit 2 equals zero (0), the least significant bits are unpacked first and the most significant bits are unpacked last.

Font Selection

In the Alphanumeric mode (bit 0 = 0), bits 3 and 4 of MCW are decoded to select one of four character fonts. The standard cursor advance associated with each font is shown in Table 4-1.

Table 4-1. Standard Font Cursor Advance

				Cursor Advance in Selected Font (number of picture elements)	
Bit No.	4	3	Font Selected	X	Y
	0	0	5 x 7	8	10
	0	1	7 x 9	10	14
	1	0	10 x 14	16	16
	1	1	16 x 15 (Programmable Font)	x-cur.adv.	y-pitch

Bits 3 and 4 have no effect in the Graphic mode (bit 0 = 1).

Memory Input Mode Selection

The Z_1 and Z_0 and V bits (bit numbers 7, 6, 5) of MCW determine how data is entered into the selected refresh memory channels. These bits are decoded to select one of four Memory Input Modes. Refer to table 4-2.

Table 4-2. Memory Input Mode Selection

MCW Bit No.	7	6	5	Memory Input Mode
Designation	Z_1	Z_0	V	
	0	0	0	OR Ones
	0	1	0	Replace Normal
	0	1	1	Replace Reverse
	1	0	0	Erase Ones
	1	1	X	Not Defined

Note that the V bit (bit 5) applies only in the Replace data mode. Any Memory Input Mode may be selected while in either the Alphanumeric or Graphic display mode. The four modes differ in the manner in which input data is conditioned prior to being written into the refresh memory.

Memory input mode applies to different commands in different ways. Note the following special rules:

a. WPX, WPXF, WPXB always use Replace Normal regardless of memory Input Mode selection.

b. BXM and XBXM always use Replace mode in graphic Pixel mode. (In Alphanumeric mode, and in graphic Word mode, BXM and XBXM use memory Input Mode.)

c. Conic instructions (vectors and circles) use Replace Normal when memory Input Mode specifies Replace Normal or Reverse.

d. Conic instructions (vectors and circles), as well as CUR write dot, use the Background Pixel Register (BPR) value when in ERASE mode. However, the BPR data is input directly as if in Replace Normal mode.

When entering pixel data into refresh memory, the low bit of the input pixel value corresponds to the low memory channel.

A discussion of each memory input mode follows.

Replace Normal

For alphanumeric data, conics and CUR 'write dot', input data is entered into refresh memory with no modification.

For conics and CUR, input data is taken from the Foreground Pixel Register (FPR) in Pixel mode; and in Word mode 1's are entered into all selected channels.

For alphanumeric data, the character is input using the FPR value in Pixel mode, or all 1's in Word mode, and the background block is input using the BPR value in Pixel mode, or all 0's in Word mode.

Replace Reverse

For conics and CUR, Replace Reverse data is entered exactly as in Replace Normal.

For alphanumeric data, Replace Reverse uses the BPR value for the character and the FPR for the backaround block, or all 0's and all 1's, respectively, if in Word mode.

ERASE 1's Mode

For conics and CUR, ERASE 1's mode enters data exactly as in

Replace Normal mode, except that input data is taken from the BPR instead of the FPR.

For alphanumeric data input, all input values are the same as in Replace Normal; that is, the character will be written from the FPR (or all 1's if in Word mode) and the background block will be written from the BPR (or all 0's if in Word mode). However, these data values are not entered directly into display memory as in Replace mode; instead, they will act upon display memory in the following manner:

The pixel value of the input data will be used as a mask on the existing data in display memory. Whenever a 1 is encountered in the input data, the corresponding bit in refresh memory will be reset to 0. Whenever a 0 is encountered in the input data, the corresponding bit in refresh memory will not be changed.

For example, suppose an area on the screen contains pixels with value 0007 (low three bits set), the FPR contains the value 0001_H, and the BPR contains the value $00C0_H$. An alphanumeric character (written in ERASE mode) in Pixel mode will have the value 0006 in the pixels which define the character, and the pixel values in the area which define the background block of the character will not be changed.

If a conic were to be drawn in ERASE mode in the above example, the pixel values in the line or circle drawn by the conic command would be $00C0_H$ (same as BPR).

OR 1's Mode

For LGE and BXM graphic word mode instructions, OR 1's mode acts in the X-Y direction. For conics, CUR, and alphanumeric characters, OR 1's mode acts in the Z direction.

In OR 1's mode, an input bit set to 1 will be entered in the corresponding position in display memory, and an input bit set to 0 will have no effect on display memory.

Since LGE and BXM graphic Word mode can use the Word mode, each pixel addressed by LGE or BXM data will either be left unchanged (pixel addressed by a 0) or will be changed to all 1's for all selected channels (pixel addressed by a 1).

For conics, CUR, and alphanumeric characters, input data is specified as in Replace Normal (i.e., FPR for conics, CUR, and character; BPR for character background block). The data is then entered into display memory in the following manner.

For each pixel, a bit set to 1 in the input data value will be entered into refresh memory and a bit set to 0 will leave the corresponding bit in refresh memory unchanged.

For example, suppose an area on the screen contains pixels with value 0007_H (low three bits set), the FPR contains the value 0001_H, and the BPR contains the value $00C0_H$. An alphanumeric character (written in OR 1's mode) in Pixel mode will have the value 0007_H in the pixels which define the character and the value $00C7_H$ in the pixels which define the background block of the character.

Similarly, a vector drawn in this area in OR 1's mode will have the pixel value 0007_H.

In OR 1's mode, as well as in Erase and Replace, Word mode automatically provides all 1's to the FPR values and all 0's to the BPR values. These are only effective values and the values which were in the pixel registers when Word mode was selected will be restored when Pixel mode is restored.

Alphanumeric Mode Cursor Advance Flag

For Alphanumeric data, this bit determines if the cursor will automatically be advanced by either the font default values (MCW bit 8 = 0) or by the value specified in the Adjustable Cursor Advance (ACA) instruction (MCW bit 8 = 1).

For Graphic display mode, this bit is not used.

SECTION 5

CURSOR

Most graphic and alphanumeric commands depend on the location of the cursor to position data display on the screen.

There are several commands (LCX, LCY, CUR) which directly control the cursor location, either absolutely (by screen coordinate) or relative to the previous position.

Another command (VCC) turns the cursor display on and off in order to save the overhead of moving a cursor on the screen when not necessary.

The cursor is automatically repositioned after certain display commands are executed (LGE, LAC, WPX, CUR, BXM). The distance the cursor is moved is given either by the ACA instruction if bit 8 of MCW is set to 1, or by the normal spacing for the currently selected font if bit 8 of MCW equals 0. (See table 4-1.)

Fifteen separate cursor locations can be stored in the 5216 Standard Firmware. These are stored and recalled by specifying a parameter set number in the Start Of Message (SOM) instruction. The number of cursor locations which can be stored is 15, regardless of the number of memory planes in the system configuration. However, only three cursors may be displayed at a time.

Changes in the cursor location caused by keyboard moves will affect only the parameter set associated with that keyboard.

Zero position of the cursor ("Home") is located at the upper left of the screen.

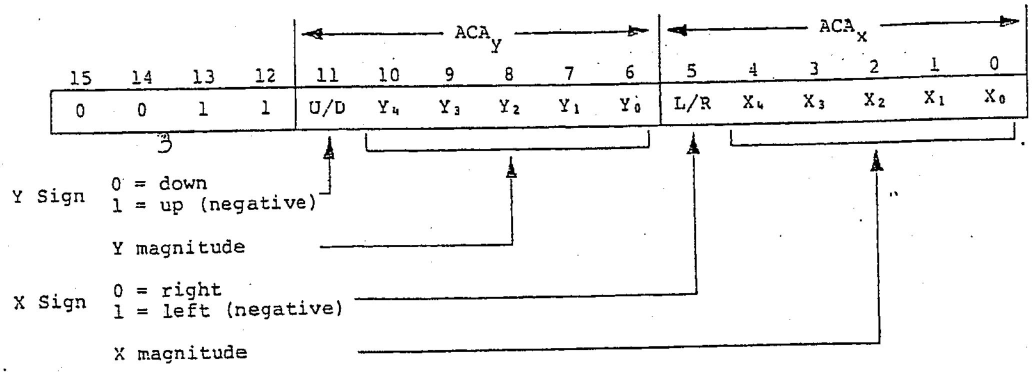
Issuing an Edit instruction to Clear (EDT) will relocate the cursor to Home. Only the currently selected parameter set is affected by cursor moves.

The cursor movement caused by the CUR instruction and by the cursor movement keys can be reversed from their normal direction by the negative direction bits in the ACA instruction.

Note

> In addition to the cursor positions addressable by the 5216 Standard Firmware instructions, each Alphanumeric Channel Set(ANCS) in the system has a cursor with its own location.

Adjustable Cursor Advance - ACA Word



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	U/D	Y_4	Y_3	Y_2	Y_1	Y_0	L/R	X_4	X_3	X_2	X_1	X_0

After an alphanumeric character is displayed, the cursor position is automatically advanced. The extent of this automatic cursor advance is determined by one of two ways: font default advance, or the Adjustable Cursor Advance (ACA) instruction.

Each character font has its own normal cursor advance distance so that after a character is written, the cursor will be advanced to the right by the width of the character. When the right viewport limit is encountered, the cursor will be advanced down by the height of the character block repositioned to the left viewport limit. This default cursor advance will automatically result in an ordinary writing sequence of characters with no spaces between characters.

In some cases, it is desirable to have extra spaces between characters, or to squash characters closer together. In these cases, the ACA instruction is used to specify the amount of cursor advance desired after each character.

The quantity given in the ACA instruction is the distance, in picture elements, that the cursor will move after each character is displayed. For example, the default cursor advance in the X direction for the 5 x 7 font is eight picture elements. If two extra pixels are desired between characters, the X operand of the ACA instruction is specified with the value of 10.

The ACA instruction cannot be used to create a diagonal path for character advance. After a character is displayed (by either LAC or BXM instructions), the cursor position is checked to determine if the right viewport limit (for positive X advance) or left viewport limit (for negative X advance) has been reached.

The ACA instruction is used in conjunction with the ACA flag bit in the Mode Control Word (MCW). The quantity of desired cursor advance is entered by using the ACA instruction; this ACA quantity will be ignored unless the ACA flag bit in the MCW is set to one (1). When this flag bit = 0, the default values are used for cursor advance after character display. When the flag bit = 1, the cursor is advanced by the amount specified in the ACA.

DISPLAY MODE	CURSOR X	CURSOR Y
Graphic MCW Bit 0 = 1 MCW Bit 8 = Don't Care	1	1
Alphanumeric MCW Bit 0 = 0 MCW Bit 8 = 1	ACA_x = 0 to 31 (as specified in the ACA instruction)	ACA_y = 0 to 31
Alphanumeric MCW Bit 0 = 0 MCW Bit 8 = 0	Font Default Values * (ACA instruction not required)	

(LGE always uses ACA_x)

* See Table 4-1 for Font cursor advance values

Cursor Positioning (Absolute or Relative) - LCX, LCY, CUR

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCX	0	1	1	1	0	R	A_x	A_x	A_x	A_x	A_x	A_x	A_x	A_x	A_x	A_x
LCY	0	1	1	1	1	R	A_y	A_y	A_y	A_y	A_y	A_y	A_y	A_y	A_y	A_y

Relative Flag

Cursor X, Y
Coordinate Value

R = 0 Cursor position is specified absolutely.

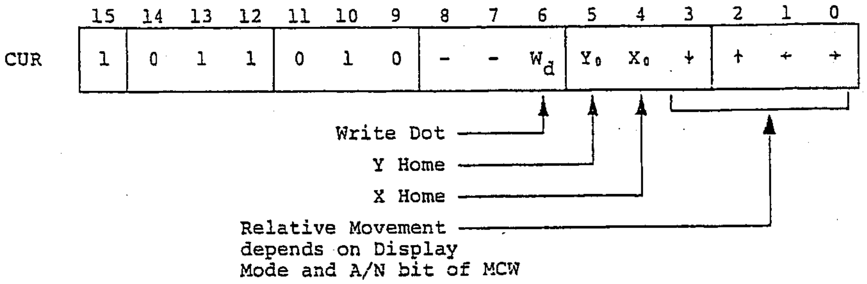
R = 1 Cursor position is specified as an increment relative to the current cursor position.

An address in the refresh memory for a cursor location is specified by the Load Cursor X (LCX) and Load Cursor Y (LCY) instructions. The least significant 10 bits of LCX and LCY are loaded into the cursor counters upon execution. If the relative flag is set, the values A_x or A_y are added to the present cursor location C_x or C_y. If the result of LCX or LCY places the cursor outside the current values values of the limits, an out-of-range error signal (code 3) is sent to the computer.

The value of the cursor position is stored in the current parameter set (as specified in the most recently issued Start Of Message command). Fifteen independent cursor locations can be stored (in fifteen parameter sets).

The value of the cursor location is given in screen coordinates. The upper left corner is (0, 0). Positive Y is down and positive X is right. The screen size is system specific.

Cursor Move - CUR



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUR	1	0	1	1	0	1	0	-	-	W_d	Y_0	X_0	↓	↑	←	→

Write Dot

Y Home

X Home

Relative Movement depends on Display Mode and A/N bit of MCW

This instruction causes the cursor to move according to the setting in the lower bits. A dot will be written at the new cursor location if bit 6 is set.

If CUR is called while MCW specifies Graphic mode, the cursor will move one pixel in the direction specified by the bit or bits set in the low four bits of CUR.

If CUR is called while MCW specifies Alphanumeric mode, the distance moved in the X and/or Y direction(s) will be dependent on the cursor advance selected by bit 8 of MCW.

If bit 8 of MCW is 0, the CUR instruction will cause the cursor to move by the default advance for the currently selected font.

If bit 8 of MCW is 1, the CUR instruction will cause the cursor to move by the amount in the X and/or Y direction(s) as specified in the ACA instruction.

The direction specified in the arrow bits of the CUR instruction may be reversed by setting a negative bit in the ACA direction bits (bits 5 and 11).

X home moves the cursor to the leftmost position of the currently selected rectangular window without affecting its Y position.

Y home moves the cursor to the top of the currently selected rectangular window without changing its X position.

The low order six bits of the CUR instruction may combine any X direction bit with any Y direction bit. For example, X home and down arrow could be set to provide a carriage return/line feed.

When bit 6 = 0, no pixel data is entered by the CUR instruction.

When bit 6 = 1, pixel data is entered into the new location of the cursor.

In Pixel mode, data is entered into all selected channels from the Foreground Pixel Register according to the Memory Input Mode selected in MCW. See Memory Input Mode of MCW for further details.

Load Index Registers - LIX, LIY

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LIX	1	0	0	0	0	0	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
LIY	1	0	0	0	0	1	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0

The least significant 10 bits of LIX and LIY specify the contents of the X and Y index registers, respectively. The index registers provide indexed refresh memory addressing for the LAC and LGE display instructions, as well as storage of parameters used in specifying conics (see the Execute Conic (EXC) instruction).

Error checking for cursor movement is done before indexing so an indexed LAC or LGE can locate the cursor outside the rectangular window.

Visible Cursor Control - VCC *EXT

Cursor Number

	15							8	7		5	4	3	2	1	0
VCC	1	1	0	1	0	0	0	1	-	-	-	-	N_2	N_1	N_0	E/D

Enable = 1
Disable = 0

This instruction causes the visible cursor to be enabled or disabled. There are 15 separate cursor positions, one for each parameter set. The number of the cursor to be controlled is specified in bits 1 to 4. Bit 0 is the enable/disable bit.

How many cursors ?

When enabled, a visible 15 x 15 crosshair is displayed, centered about the current X, Y coordinates of the computer cursor. The pixel data at the cursor location is automatically stored when the cursor is enabled and when the cursor is moved or disabled the stored data is replaced.

Note

The A/N channel cursor is not affected by the VCC instruction.

Standard Firmware includes three independent visible cursors, each of which may be enabled or disabled independently.

The three cursors are associated on Power Up with parameter sets 1 through 3.

Power Up Cursor Association

Cursor	Parameter Set
1	1
2	2
3	3

Each of the 15 parameter sets has a "context switch" which can be used to change the association of the cursor with its parameter set.

Parameter Set (Decimal)	Context Switch (Hex)
1	2
2	4
3	6
4	8
5	A
6	10
7	12
8	14
9	16
10	18
11	1A
12	1C
13	1E
14	20

Each cursor has an address in 5216 memory which can be loaded to associate that cursor with the parameter set specified.

Cursor	Memory Location
1	E08C6
2	E08C8
3	E08CA

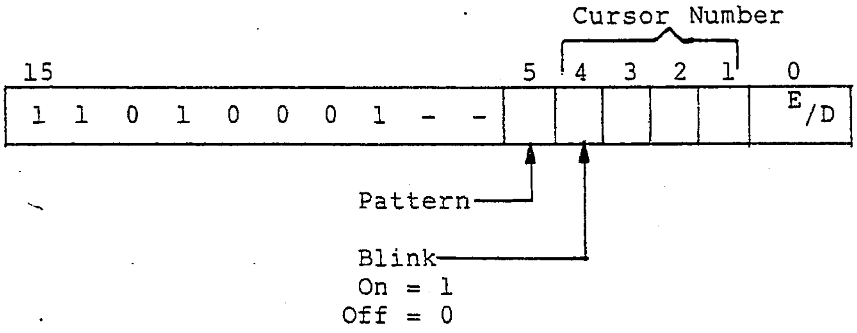
The Load Cache Buffer (LCB) instruction is used to place data into 5216 memory.

For example, to associate cursor number 1 to parameter set 3, cursor number 2 to parameter set 2, and cursor number 3 to parameter set 11, load the following:

Address	Data
E08C6	6
E08C8	4
E08CA	1A

Hardware Cursor Module - Optional (8 cursors)

The optional Hardware Cursor Module supplies eight independently controlled cursors. Each cursor may be enabled or disabled and made to blink or not to blink, and each cursor may be displayed in either one of two display patterns. The first pattern (pattern bit = 0) is a 16 x 16 crosshair; the second pattern is a 16 x 16 pixel box inside a 32 x 32 pixel box.



The cursors displayed by the Hardware Cursor Module are associated at Power Up with parameter sets 1 through 8. These default associations may be changed in exactly the same way as described above for the three firmware cursors. By loading the eight addresses, starting from E08C6, with the appropriate context switches for each parameter set as listed above, the parameter set with which the cursors are associated may be programmed.

The memory locations which are loaded to reassign cursor association with parameter sets are:

Cursor	Address (Hex)
1	E08C6
2	E08C8
3	E08CA
4	E08CC
5	E08CE
6	E08D0
7	E08D2
8	E08D4

Take for example, the following association of cursors with parameter sets:

Cursor Number	Associated Parameter Set
1	11
2	6
3	3
4	13
5	1
6	2
7	9
8	14

To create this set of associations, load the following data into the 5216, starting at address E08C6 and using the Load Cache Buffer (LCB) instruction:

Address	Data
E08C6	1A
E08C8	10
E08CA	6
E08CC	1E
E08CE	2
E08D0	4
E08D2	16
E08D4	20

SECTION 6

ALPHANUMERIC DATA DISPLAY

There are two ways of entering alphanumeric data into the 5216 Standard Firmware for display. One is by the Load Alphanumeric Character (LAC) instruction and the other is by Block Transfer Mode (BXM and XBXM).

LAC sends one character code per instruction, whereas BXM sends many characters per instruction in rapid Direct Memory Access transfer.

Both alphanumeric instructions require the MCW bit 0 to be reset to 0. The font currently selected in the MCW will be displayed. The MCW is also used for BXM unpacking order.

Both LAC and BXM instructions display characters at the current cursor location (or indexed location) and after each character is written, the cursor is advanced. Cursor advance is as given in table 4-1 when the MCW bit 8 is 0. When MCW bit 8 is set to 1, cursor advance is given by the values specified by the programmer in the ACA instruction.

Font size provided by 5216 Standard Firmware are 5 x 7, 7 x 9, and 10 x 14.

A user programmable font can also be loaded using the Load Programmable Font (LPF) instruction and then written by LAC or BXM in the same way as the firmware supplied fonts.

The above alphanumeric display capabilities are implemented through the 5216 Standard Firmware. Pixel data for these characters is stored in the memory channels available in the system configuration. The same color capabilities and channel selection features are used for alphanumeric display as are used for graphic display.

Note

> In addition to the above mentioned alphanumeric display, the 5216 Display Computer can contain as a hardware option an alphanumeric display capability on a separate board with its own processor and refresh hardware. The ANCS can drive a separate monitor to display alphanumeric data with complete independence from the refresh memory. ANCS programming is done by a separate instruction set communicated from the 5216 Standard Firmware instruction set by the Select Device (SDEV) instruction. See ANCS documentation and Section 15 for SDEV instruction.

The ANCS can also be selected by means of the Alpha-Keyboard function key on the 5216 Standard Firmware function key set.

Load Alphanumeric Character - LAC

	15					10	9	8	7	6	5	4	3	2	1	0
LAC	1	0	0	1	1	0	I	-	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1

Indexed Address Flag

Character Code

The eight least significant bits of LAC specify one of up to 256 possible character codes, depending on the selected font. Only the font specified in MCW will be written. Limit checks are performed before the character is written and the cursor is advanced after the write operation has been performed. Bit 9 of LAC permits indexed addressing. MCW must specify the Alphanumeric mode when LAC is used. Characters may also be written in the Block Transfer Mode (BXM). Character codes are listed in Appendix B.

For indexed addressing (bit 9 = 1), the starting address for the write is $C_x + I_x$ and $C_y + I_y$, where C is the cursor coordinate and I is the index register contents as loaded in the Load Index Registers (LIX, LIY) instruction.

Each character is set within a block. When the Memory Input Mode (bits 5 - 7 of MCW) is set to Replace Normal, pixel input data for the character is taken from the Foreground Pixel Register and the block (background pixel input data is taken from the Background Pixel Register (see Load Pixel Value Registers - LPXF, LPXB).

In Replace Reverse mode, the character input data is taken from the Background Pixel Register (BPR) and the block input data is taken from the Foreground Pixel Register (FPR).

For ERASE 1's mode and OR 1's mode, input data is the same as Replace Normal but it is logically acted upon as specified by the input mode.

In Word mode (MCW bit 1), the data input from the FPR will automatically be set to all 1's and the data input from the BPR will automatically be set to all 0's. (However, the contents of the pixel registers will not be lost by changing from Pixel to Word mode.) Word mode is time efficient for character generation. However, color data is limited in Word mode to all 1's and all 0's.

When a character overlaps the boundary of the window limits in Word mode, the character will be aborted; in Pixel mode, it will be clipped at the boundary.

Associated with each character font is a fixed character block area within which the character specified by LAC is written. Character placement within the block is shown for each font in figure 6-1. The cursor position at the start of LAC is the top left picture element of the character block area, and the character is written one line at a time from top to bottom.

If different adjacent character spacings are required, the ACA flag of MCW (bit 8) may be set and the desired spacings specified by an ACA instruction. Note that the character block sizes shown in figure 6-1 correspond directly to the standard cursor advances shown in table 4-1. Character set diagrams and codes are provided in figures 6-2 and 6-3 and Appendix B.

Load Programmable Font - LPF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	0	0	1	0	1	-	-	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1	LPF

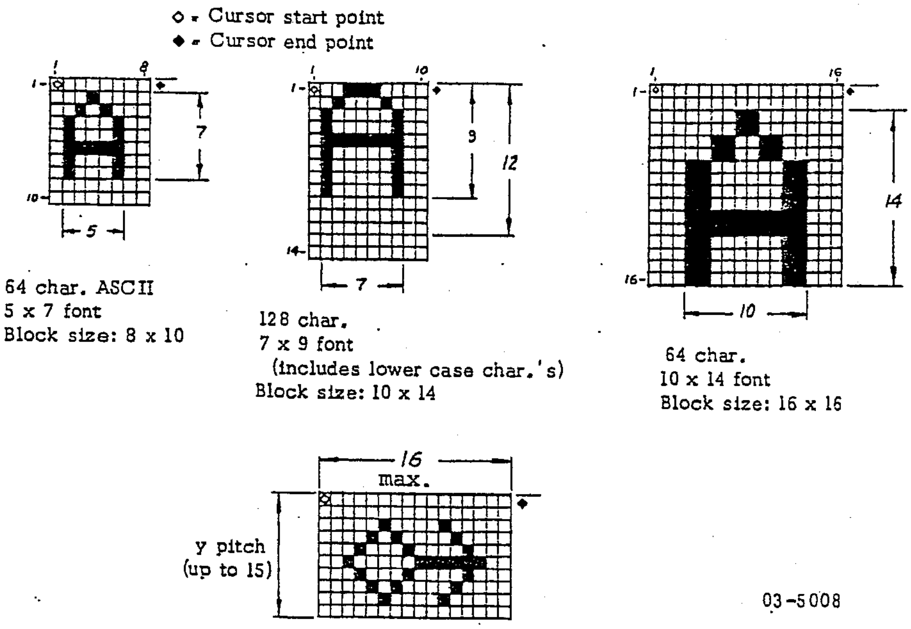
0	0	X_4	X_3	X_2	X_1	X_0	Y_3	Y_2	Y_1	Y_0	-	X_3	X_2	X_1	X_0	X, Y Pitch Word
		X Cur ADV.					Y Pitch				X Pitch-1					

D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Character Data (next 15 consecutive words)

The 5216 Standard Firmware instruction set supports a user programmable font of up to 256 characters.

Load Programmable Font (LPF) is a multiple-word instruction used to specify any one of 256 characters in the programmable font. The 16 words which follow LPF are stored in sequential locations of the programmable font RAM, starting at the address specified by bits 0 through 7 of LPF. To load all 256 characters, it is necessary to issue 256 different LPF instructions, each followed by 16 data words in the format shown above.

Each programmed character may have a Y dimension of 2 to 15 elements, and an X dimension of 1 to 16 picture elements as specified in the X and Y pitch fields. Note that the X pitch field specifies one less than the true X pitch. Cursor advance information is also specified by the pitch word. After a programmed



64 char. ASCII
5 x 7 font
Block size: 8 x 10

128 char.
7 x 9 font
(includes lower case char.'s)
Block size: 10 x 14

64 char.
10 x 14 font
Block size: 16 x 16

256 char.
16 x 16 programmable font
Block size: x = x pitch
y = y pitch

Figure 6-1. Character Block Sizes

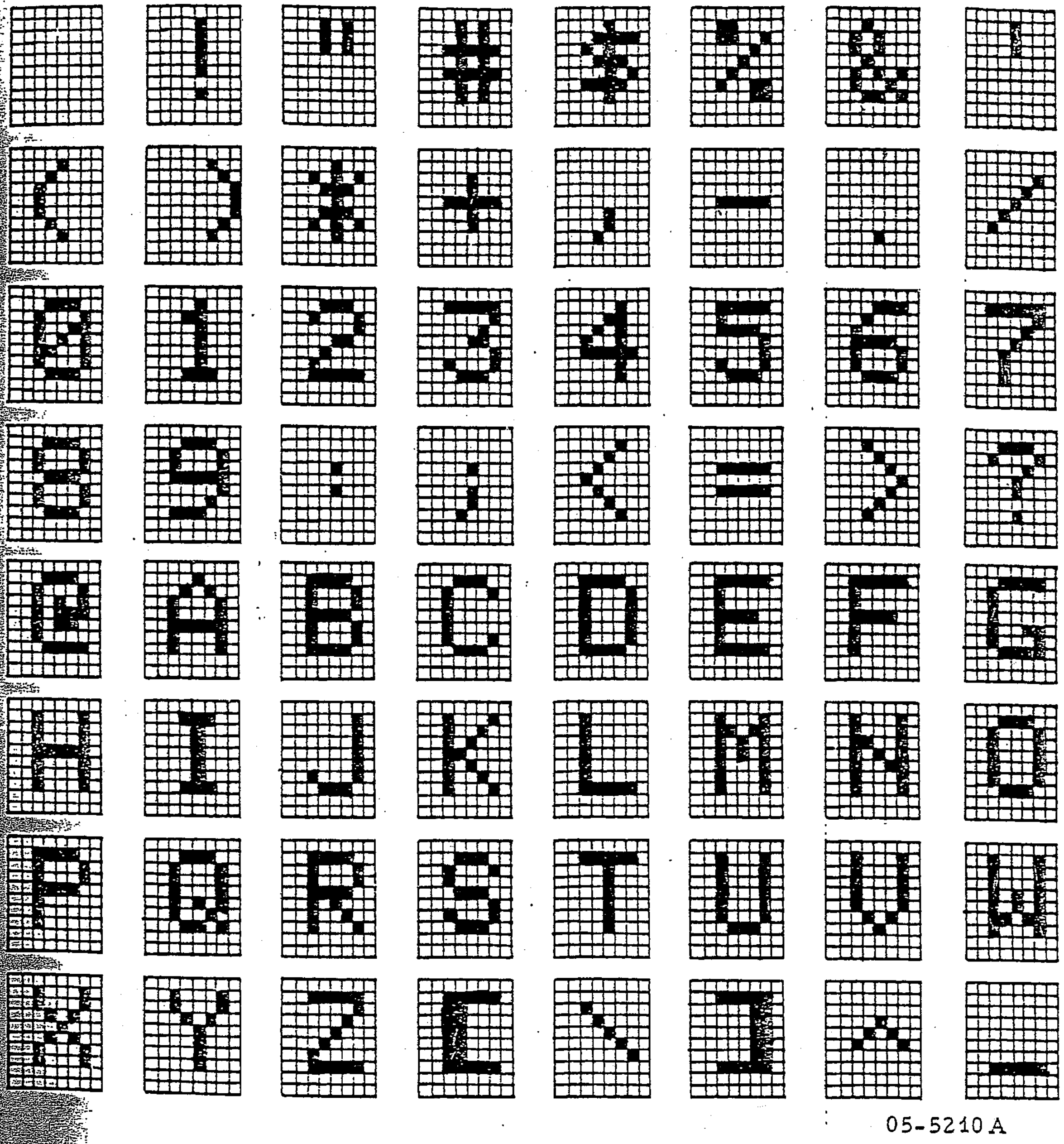


Figure 6-2. 5 x 7 Character Set

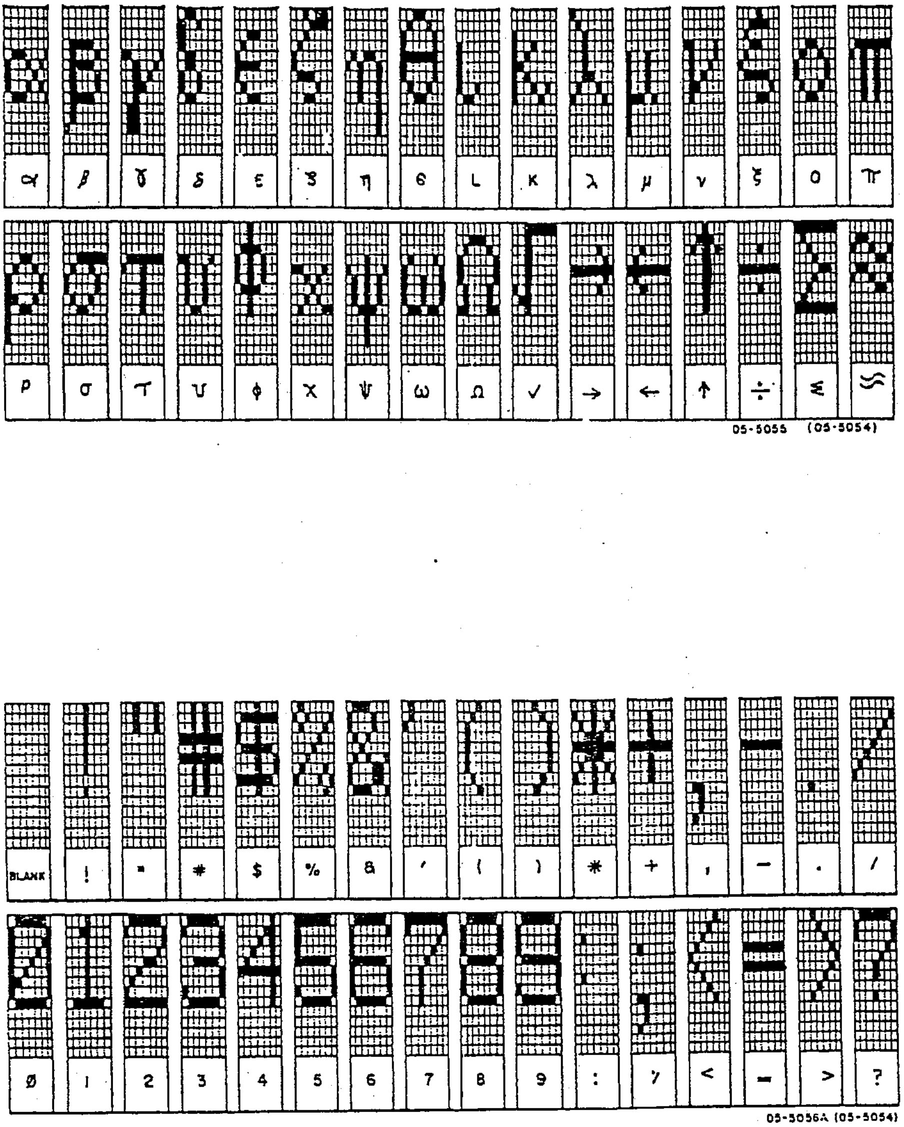

α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π
ρ σ τ υ φ χ ψ ω Ω ✓ → ← ↑ ÷ ≤ ≈
05-5055 (05-5054)
BLANK ! " # $ % & ' ( ) * + , – . /
Ø 1 2 3 4 5 6 7 8 9 : ; < = > ?
05-5056A (05-5054)

Figure 6-3. 7 x 9 Character Set
(Sheet 1 of 2)

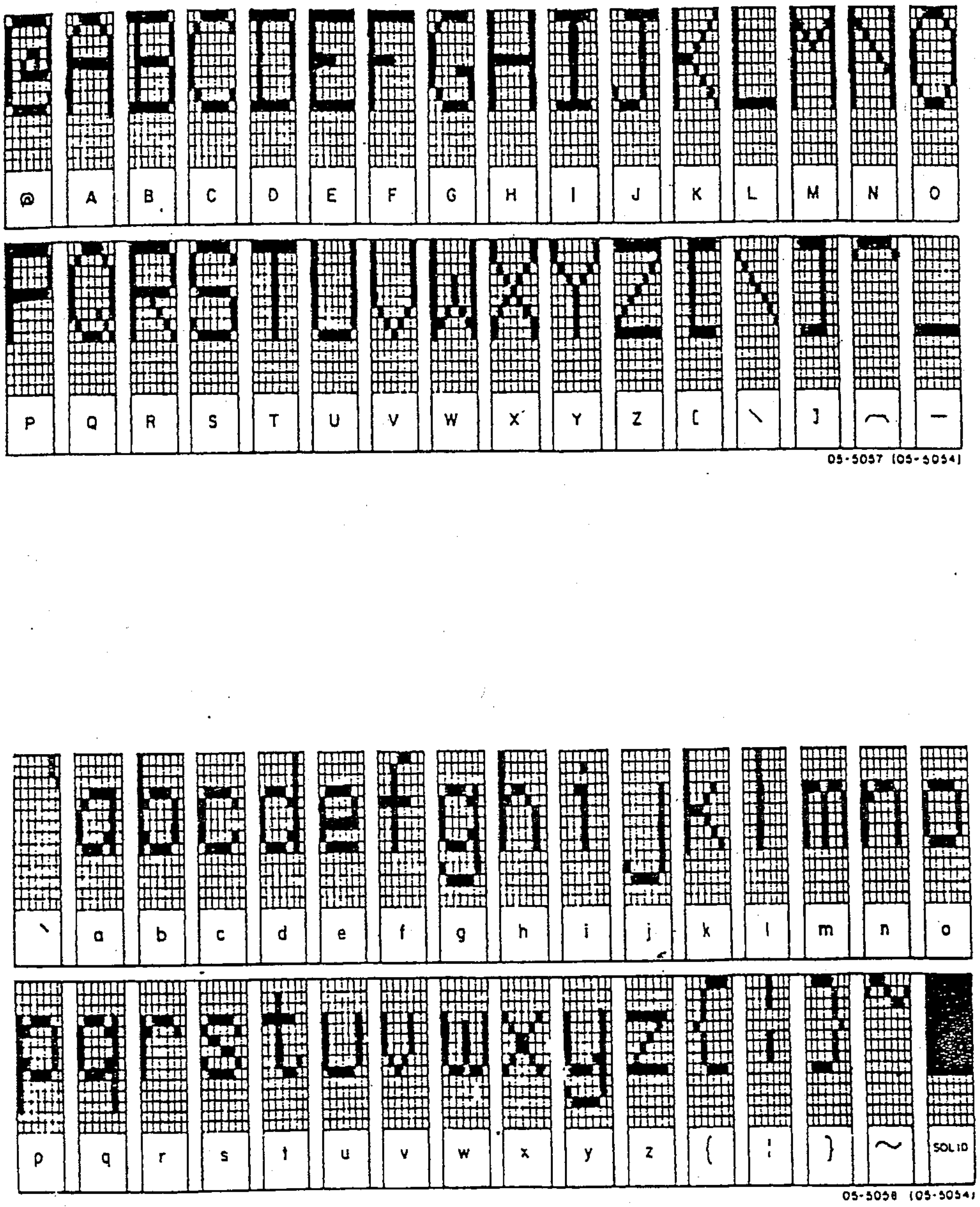

@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [\ ] ⌒ —
05-5057 (05-5054)
` a b c d e f g h i j k l m n o
p q r s t u v w x y z { ¦ } ~ SOLID
05-5058 (05-5054)

Figure 6-3. 7 x 9 Character Set
(Sheet 2 of 2)

character is written, the X field determines the amount of cursor X advance, and the Y pitch field determines the cursor Y advance when a new line is begun. The Y pitch field also specifies the character block Y dimension. Note that when loading the font, LPF must be followed by a pitch word and 15 data words, even if the programmed character occupies less than 15 scan lines. When the character is displayed, only the number of lines specified in the Y pitch field are displayed, and only the number of bits will be displayed as are specified in the X pitch (starting with the least significant bit).

The 15 words of character data (D) must be specified such that the least significant bit of data (bit 0) will appear at the left side of the displayed character. The first word specified will be displayed at the top of the character.